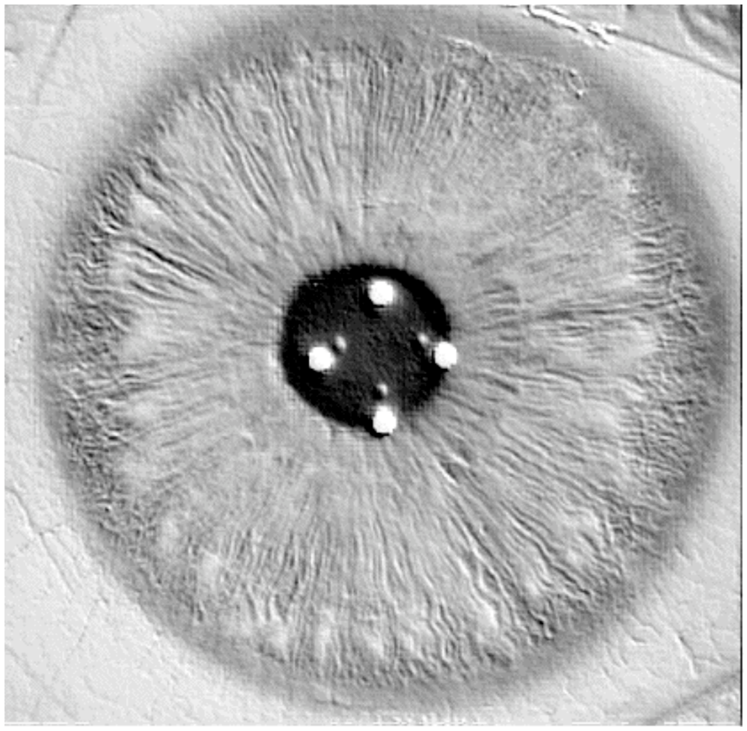
18-551, Spring 2003
Group #6, Final Report

THE EYES HAVE IT:



Iris Identification

Steve Lee(cslee@andrew.cmu.edu)
Jeff Mrenak (jm73@andrew.cmu.edu)
Lynna Quandt (lhq@andrew.cmu.edu)

Table of Contents

The Problem

Security is a huge issue facing corporations and governments today. From attempting to identify credit card users for on-line purchases to ensuring corporate and government secrets are kept secret, new forms of security have been emerging rapidly. Since passwords can be forgotten or stolen, the world of biometrics has entered the security arena to increase security. This includes the fields of fingerprint, facial, voice and iris verification.

Many times, companies or governments need to discern if a person requesting access to information or an area is in fact the person they claim to be. In these situations, the user is cooperative and it can take a few seconds for the system to identify the user. Therefore, one does not need to be overly concerned with lighting or unusual appearances as the person knows that they are using an identification system.

For the purpose of our project, we have set up the following scenario. A company is working on a product development project and is worried about the security of its files. They believe it to be very valuable and have had past incidents of theft by competitors. Therefore, they have chosen to install an iris identification system. Fifteen employees have been granted access to the files. A previous scan of each employee's iris has been taken and is stored in a database. The users are cooperative with the system and a scan is taken by looking into a device with a camera mounted on the inside. The company is willing to tolerate an occasional rejection of an authorized user as long as unauthorized users are not accidentally accepted.

Why Iris Identification?

Unlike other biometrics such as fingerprint or facial recognition, the iris does not change over time. Fingerprints can be interrupted by scars, burns, or blisters. Additionally, when a fingerprint is taken, the skin can be stretched, thus distorting the image. Facial recognition also has its problems. People's faces change as they age. Therefore images are not useful for long periods of time. Lighting, expression and obstructions such as sunglasses or scarves can make facial recognition less accurate. Iris identification does not have these problems. The iris does not change as people age nor can its features be interrupted. Below is a chart detailing different biometrics and their performance. (Group 3 –1999, "A Study in Iris Recognition, Final Report")

	Iris	Retina	Fingerprint	Voice	Facial
Accuracy/Cross over Error	.00008%*	1.5%	5%	5-10%	2.5%
Average Decision Time	2-3 sec	3 sec	5 sec	5-13 sec	2.5 sec
Non-Invasive Process	Yes	No	No	No	Yes
Data Acquisition Without Contact	Yes	Yes	No	No	Yes
Performs Verification	Yes	Yes	Yes	Yes	Yes
Performs Identification	Yes	Yes	Limited	No	Limited
Fraud Susceptibility	Low	Low	Medium	Medium	Medium
Uses Scientific Unique Characteristic	Yes	Yes	Yes	No	No

Data File Size (bytes0	512	96	900	3,000	4,000
User Friendliness	High	Low	Medium	Medium	High

* There has never been a false accept/false identification; therefore this is a calculated theoretical value

What has been Done Before?

John Daugman is the father of the iris recognition systems used today. His research and codes have been licensed to Iridian Technologies and sold to such companies as Panasonic, EyeTicket, LG, and IBM - Schiphol Group. He began the recognition process by first locating the iris in the scan. Upon finding it, the scan is then converted into doubly dimensionless projected polar coordinates. This allows for compensation for alterations to the scan due to squinting or pupil dilation. Once this conversion has been done and the iris is divided into four quadrants, quadruple 2D wavelets are done to extract phase information. The phase code is then used to extract a Hamming distance sequence. Then, based on a predetermined deviation limit, the sample will be accepted or rejected. (Daugman, "How Iris Recognition Works")

A 1999 551 project is the only other one with regards to iris recognition. They chose to follow the method laid out by Daugman. To gather pictures of irises, they developed a camera unit to take their own pictures. This proved to be a large problem for the group though. They were not able to get good pictures because their camera did not have autofocus and it took a very long time for them to get a good image. The time required was longer than most subjects were willing to sit for. However, once they did get an image, they used Daugman's method. However, they did not get the program to run on the EVM. Additionally, they found that their method was not very accurate. (Group 3 –1999, "A Study in Iris Recognition, Final Report")

Our Goal

Our goal is to use iris identification to determine if a person should be given access to a secure system. Each person who has access to this system has previously submitted an iris scan. We will designate fifteen people from our database as those who should be given access to the system. Our identification system will accept a given scan as an input. It will then determine if this scan belongs to a person who has access to the system or not. If the scan does match an authorized user, it will identify who the user is. If not, the user will be rejected.

Iris Recognition Process Overview

Our iris recognition process required several steps. First, we needed acquire an image of an eye. In this case, we used images contained in a database stored on the PC. Second, we needed to find the iris within the image of the eye. In our implementation, this was done on the PC since it is quick enough to handle the processing load and it saves from having to send much larger chunks of data to the EVM. Third, we needed to extract the iris from the image, and store it in a format that was invariant among many runs of the process and across all possible eye images. We also accomplished this on the PC, and we sent the resulting iris information to the EVM for processing. Fourth, we needed to extract the features of the iris and use them to build an iris code of uniform length across all irises, irrespective of image size or apparent level of detail. This we did on the EVM, taking advantage of the EVM's ability to quickly apply filters across substantial buffers. Next, we used the EVM to quickly compare this code against an existing database of codes using Hamming distance as a measure of similarity. Finally, we sent the resulting list of Hamming distances back to the PC for the application of an acceptance criterion to determine if a similar enough iris was found in the database.

Image Acquisition

Before any processing can begin, we need to have an image to work with. In production systems, this is accomplished with a camera, sometimes using near infrared light.

In 1999, the 18-551 group attempting iris recognition decided to use a Logitech QuickCam VC to capture eye images. This noble decision would haunt them. They spent considerable time and energy just trying to get usable images, and they were never quite sure if their low image quality was foiling all their subsequent efforts.

Since our main goal going into this project was to become intimately familiar with the process of iris feature extraction, we felt that fighting through the problems that vexed the previous group was somewhat off topic and an inefficient use of limited resources. We decided to learn from their troubles and bypass the low-end camera quagmire. Of course, this left us with the problem of having no eye images.

The Database

Professor Kumar came to our rescue. He possessed and released to us a database of eye images stored in Windows bitmap format. The database contained color images of the left and right eyes of 24 different people. For our purposes, this was equivalent to 48 unrelated eyes since the fine textures of two genetically related irises are no more alike than two that are genetically unrelated. Each image contained the sclera, iris, and pupil. It was clear that the subjects used for the database had their eyes held wide open since there were no obstructions from eyelids or eyelashes. After off-line reductions and gray scaling, these images contained some 100 pixels in radial resolution across the iris. This was well above the 70-pixel requirement specified by Daugman. The only obvious problem with the images was the four bright white illumination reflections present in every image. In nearly every image, these reflections crossed the pupillary boundary of the iris. Other than that one flaw, these images were ideal.

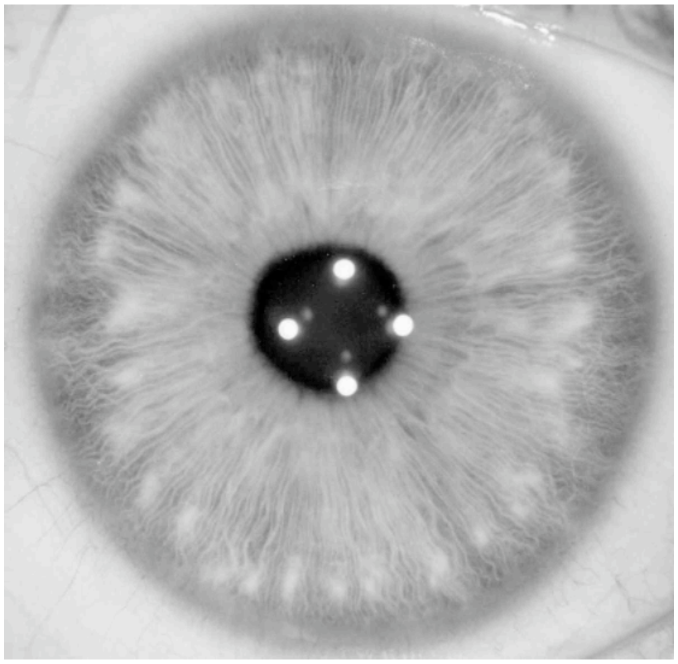


Figure 1 – Sample iris image

Our only task then in image acquisition was to import the bitmap files, and convert every pixel's RGB values to the integer pixel intensity $0 \le I \le 255$ required for processing. This conversion can be done in many ways. We decided to use the method and weightings specified by the National Television Standards Committee (NTSC) (see Equation 1). Although we originally downloaded some code to import bitmaps (cannot find the site anymore), the code did not work for our grayscale images. In the end, we completely rewrote the importing routine, although some simple support routines remained unchanged, e.g., the routine to swap bytes and words depending on the endianess of the host machine.

$$I = 0.299R + 0.587G + 0.114B$$

Equation 1 - NTSC conversion formula

Iris Localization

In Daugman's system, iris localization begins with determining if an iris is visible within the sample image. Since we were working from ideal images of eyes and not from image captures at a distance, we decided to assume that an iris was visible in all images we would process. With this assumption in hand, our remaining localization tasks were to locate the boundary between the iris and the sclera, and to locate the boundary between the iris and the pupil.

To do this, Daugman's describes a continuous-form equation for circular edge detection. This theoretical detector takes advantage of the nearly circular geometry of the iris (i.e., both boundaries are very nearly circles.) As can be seen from this equation (see Equation 2), Daugman recommends searching for the maximum partial derivative with respect to radius of a normalized contour integral of the image along circular arcs. Convolution with the Gaussian smoothing function is used to set the scale of analysis, allowing this generalized detector to first detect the iris-sclera boundary and then the iris-pupil boundary.

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r}\, ds \right|$$

Equation 2 - Continuous form of Daugman's circular edge detector

Of course, our PC operates in the discrete world and so Equation 2 cannot be directly implemented. Fortunately, Daugman provides a discretized version (see Equation 3) that can be used for locating the iris-sclera boundary.

$$\max_{(n\Delta r,x_0,y_0)} \left| \frac{1}{\Delta r} \sum_k \left\{ \left(G_\sigma((n-k)\Delta r) - G_\sigma((n-k-1)\Delta r)\right) \sum_m I\left[(k\Delta r\cos(m\Delta\theta)+x_0),(k\Delta r\sin(m\Delta\theta)+y_0)\right] \right\} \right|$$

Equation 3 - Discrete operator used to find iris-sclera boundary

In Equation 3, Daugman uses the discrete approximation of the partial derivative, he replaces continuous convolution and integration by discrete sums, and he swaps the order of some operations for efficiency of implementation. In actual practice, Daugman suggests that integral summation not be performed around an entire circle, but rather along two circular arcs lying within opposing 90^o cones centered on the horizontal meridian. This eliminates the problem of possible obstruction of the sclera boundary by the eyelid on the top and bottom of the iris.

We could find no publicly available software that implemented Equation 3, so we wrote the code ourselves. (In subsequent discussion with Professor Casasent, it became clear that we might have been performing a far too narrow search for available edge detection routines.) In testing on the PC, we found that this detector worked quite well. It did not always find the sclera boundary down to the pixel, but it came pretty close. In retrospect, these small errors were inevitable since our images were large and detailed and in many images it was obvious that the iris boundaries were not exactly circular. It seems likely that Daugman uses unpublished and much more sophisticated detection algorithms in his commercial software. For our purposes, the results of the edge-detecting algorithm were satisfactory in all cases except 1, image 20R. In this case, our algorithm clearly missed the center and radius of the sclera boundary. Possible reasons will be discussed later.

In our implementation, we found it useful to limit our search for the center of the boundary circles to about 50% of the pixels in each axis centered on the exact middle of the picture. In this way, we cut out 75% of the potential centers, thus eliminating much unnecessary processing. We were able to achieve this reduction in resources since we knew that in all of our images, the eye was at least close to center.

Likewise, our implementation checks that all points in a contour integral actually lie in the image before beginning the summation. Again, this check gives us some reduction in the necessary computations for finding the best radius for any given center. The price we pay is that it is unlikely if not impossible that we will find the sclera boundary for any iris in which all of the boundary within the opposing cones of analysis lies on the image. To wit, if the left or right portion of the sclera boundary is missing from the picture, our algorithm will not find the proper center and radius of that boundary. We felt justified in taking advantage of these computational savings since our database contained no images in which any part of the sclera boundary was missing.

In actual practice, Daugman recommends using Equation 3 only for determining the iris-sclera boundary. Since the pupillary boundary is generally much less pronounced than the boundary with the sclera, Daugman describes a slightly different version of circular edge detector (see Equation 4) for locating this boundary.

$$\max_{(n\Delta r, x_0, y_0)} \left| \sum_k \left\{ \frac{\left(G_\sigma((n-k)\Delta r) - G_\sigma((n-k-1)\Delta r)\right) \sum_m I\left[(k\Delta r\cos(m\Delta\theta)+x_0),(k\Delta r\sin(m\Delta\theta)+y_0)\right]}{\Delta r \sum_m I\left[((k-2)\Delta r\cos(m\Delta\theta)+x_0),((k-2)\Delta r\sin(m\Delta\theta)+y_0)\right]} \right\} \right|$$

Equation 4 - Discrete operator used to find iris-pupil boundary

We can see from Equation 4 that Daugman simply adds a denominator to Equation 3. This denominator takes contour integrals at slightly smaller radii than the numerator. In so doing, it exploits the biological fact "that the interior of the pupil is generally both homogeneous and dark". (Daugman, "How Iris Recognition Works") Essentially, this denominator becomes very small and stable when it is operating over the interior of the pupil, generating a small divisor and thus a much more pronounced and detectable maximum.

Again, we found no publicly available code, and so we wrote our own. But, we had serious problems getting this detector to work on the PC. As discussed earlier, each of our images had four bright white spots caused by a reflection of the illumination used for capturing the image. These spots were always located in the pupil and in most cases extended across the border. These wreaked havoc with our detector, as it foiled not only the numerator's attempt to find a maximum in the gradient denoting the boundary, but they also killed the denominator's attempt to exploit the dark regions of the pupil. After much time and consternation, we decided to hack around the problem and return to it later if time allowed; it did not. Our hack was to simply use the center of the circle defining the sclera boundary as the center of the pupillary boundary. This is not necessarily valid as the iris-sclera and iris-pupil boundaries are frequently not concentric. Further, we used a fixed radius for the pupillary boundary that we determined from examining the images to be about 50 pixels. Again, this is not completely valid as the pupillary boundary moves over time, although surely not in our images, and not all boundaries were exactly 50 pixels.

The net result of these hacks was that in some images we were losing a portion of the iris and in others we were including a portion of the pupil. Neither one of these problems is fatal, but they must necessarily reduce the randomness or degrees of freedom that we capture.

At our demo, Professor Casasent suggested a fix for this problem of filtering out the specific values for these white spots and then performing our edge detection. This was a great solution. Unfortunately, it did not occur to us during the semester. In hindsight, it is clear that had we brought this problem to the Professor's attention during the semester, we would have saved much time and trouble, and built a slightly better mousetrap.

Iris Extraction

At this point in the processing of our sample image, we had the center and radius of the iris-sclera boundary and the center and radius of the iris-pupil boundary, or at least our best guesses at them. Our next task was to extract the iris in some way, and to store the iris data in a format that would be invariant across program runs and across different images.

Doubly Dimensionless Projected Polar Coordinate System

Our sample image is stored using the familiar *x* and *y* coordinates of the Cartesian system. The nearly circular geometry of the eye is clearly more suited to the polar system. And if we simply pulled the iris pixels out one by one and stored them in a buffer, even in polar form, it is clear that the size of our buffer would depend on our sample image size or apparent level of detail. Further, in the case of pupil variations in a given iris, our pixel coordinates would not be adjusted in a manner conformant with the actual way the structures of the iris adjust. And so we turned to the doubly dimensionless projected polar coordinate system as a solution to all of these issues. Since this system is polar it more naturally conforms to the eye geometry. And since this system is dimensionless, it solves our problem of differing buffer sizes. Further, the actual behavior of the iris structures have been found to be modeled well by this coordinate system. Specifically, it has been found by Daugman that the structures of the iris move with pupillary variations as if the iris was a homogeneous rubber sheet, "having the topology of an annulus anchored along its outer perimeter, with tension controlled by an off-centered interior ring of variable radius". (Daugman, "How Iris Recognition Works") In other words, the structures move with pupillary radial variations as if they were anchored at their boundary with the sclera and stretched from there by the dilation and contraction of the pupil.

The transformation to this system from the Cartesian system is accomplished using Equations 5 (see below).

$$
\begin{aligned}
x(r,\theta) &= (1-r)x_p(\theta) + r x_s(\theta) \\
y(r,\theta) &= (1-r)y_p(\theta) + r y_s(\theta)
\end{aligned}
$$

Equation 5 - Equations for the transformation from Cartesian to doubly dimensionless polar coordinates

These equations determine the *x* and *y* for every *r* and θ in the iris image. Note that x_p, x_s, y_p, and y_s denote the computed *x* and *y* boundary coordinates for the iris-pupil and iris-sclera boundaries respectively. Also note that θ expressed in radians is always dimensionless, whereas the radius *r* in this system is normalized to [0,1] thus making it dimensionless as well. To actually perform this sampling and transformation in a discrete manner, we had to choose an appropriate number of samples that captured as much iris information as possible without duplicating pixels. Based on our image size and resolution, we found that approximately 100 pixels radially and 300 pixels angularly were good choices that met these criteria. So, after our extraction and transformation to dimensionless coordinates, we were always left with a uniformly sized iris buffer of approximately 30k. Note, all code used to perform iris extraction was original, though clearly based on Daugman's methods.

Building the Iris Codes

In Daugman's system, the iris code is built by projecting the image of the iris onto 2-D Gabor filters (see Equation 6) to extract the iris's phase information at multiple scales of analysis. Note that amplitude information is not used because it has been found by Daugman that it is not very discriminating, and depends heavily on extraneous factors. Both the real and imaginary members of the filter are used, so we get a complex valued result for each region of analysis at each scale of analysis.

$$
G(x,y) = e^{i\omega(\theta-\theta_0)} \cdot e^{\left(\frac{-(r-r_0)^2}{\alpha^2}\right)} \cdot e^{\left(\frac{-(\theta-\theta_0)^2}{\beta^2}\right)}
$$

Equation 6 - 2-D Gabor filter used for iris feature extraction

It is not clear from his published works how Daugman partitions his sample irises for analysis, or with how many scales of analysis he works. So, we decided, somewhat arbitrarily, to partition our sample iris image into 8 radial and 8 angular segments, thus breaking the iris into 64 equally-sized pieced of about 12 pixels by 32 pixels. We also decided to perform our Gabor analysis at 2 scales of analysis. Although these numbers correspond to a smaller code and fewer degrees of freedom than Daugman's, they seem reasonable given the small size of our database and the relatively small accumulation of error probability.

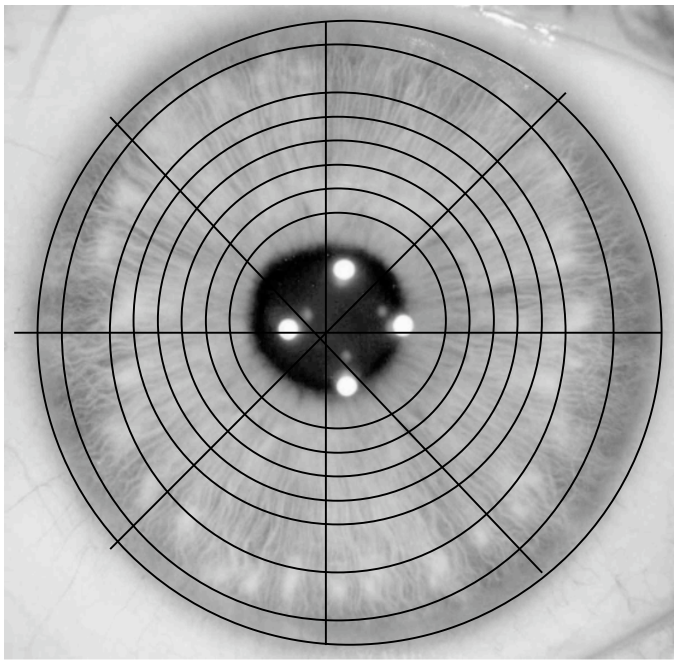


Figure 2 – Segmentation of iris image

For each region of analysis, we apply the filter to the image once for each scale of analysis. Out of each filter application we get a real number and an imaginary number. We quantize this complex result to one of the four quadrants of the complex plane by assigning 1s and 0s to a complex bit *h* in a manner which gives rise to a gray code of length 2. We use the gray code method to get a little better error performance for free. So, our problem comes down to application of the Gabor equations to small regions of our image (see Equation 7).

$$h_{\mathrm{Re}} = 1 \; if \; \mathrm{Re} \int_{\rho}\int_{\phi} e^{-i\omega(\theta_0-\phi)} e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} I(\rho,\phi)\,\rho\, d\rho\, d\phi \geq 0$$

$$h_{\mathrm{Re}} = 0 \; if \; \mathrm{Re} \int_{\rho}\int_{\phi} e^{-i\omega(\theta_0-\phi)} e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} I(\rho,\phi)\,\rho\, d\rho\, d\phi < 0$$

$$h_{\mathrm{Im}} = 1 \; if \; \mathrm{Im} \int_{\rho}\int_{\phi} e^{-i\omega(\theta_0-\phi)} e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} I(\rho,\phi)\,\rho\, d\rho\, d\phi \geq 0$$

$$h_{\mathrm{Im}} = 0 \; if \; \mathrm{Im} \int_{\rho}\int_{\phi} e^{-i\omega(\theta_0-\phi)} e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} I(\rho,\phi)\,\rho\, d\rho\, d\phi < 0$$

Equation 7 - Equations to set the complex bit h using Gabor filters

In these equations, ρ and ϕ are radius and angle parameters that range over the current region of analysis, and r_0 and θ_0 denote the center of this region. ω, α, and β denote the Gabor wavelet frequency, radial length, and angular length respectively. Daugman does not describe in any detail how he sets these parameters. So, we experimented until we got what we thought were reasonable results. In our code these values were computed based on the sampling interval and region size. For the first scale of analysis, they were set to $\omega \approx 32.4 radians$, $\alpha = 0.01$, $\beta \approx 0.021$. Note that α and β are dimensionless and correspond to a fraction of the region size. For our second run, at a smaller scale of analysis, we stepped ω up one octave and halved the values for α and β.

We had 64 regions of analysis, 2 scales of analysis, and 2 bits of information per application of the Gabor filter. So, we ended up with $64 \cdot 2 \cdot 2 = 256$ bits of information per iris. Since our data sizes were so small and the time to transfer and process iriscodes is negligible for our purposes, we decided to use a byte to represent each bit, rather than worry about bit packing and implementation storage issues. We paid a price of a factor of 8 increase in the size of our data, but this equated to only 256 bytes per iriscode.

As the filter applied to each region at a particular set of size and frequency parameters is identical, we really need to build only four filters total, one real and one imaginary at each of two scales of analysis. Since these do not depend on the image, they can be built offline and simply transferred to the EVM on program startup. We mimic this behavior by simply building the filters, using all original code, on the PC at startup as if they already existed, and then transferring them to the EVM before the user has any interaction with the system. The time to transfer is not noticeable to the user.

Code Comparison

For comparison to our just-computed sample iris code, we needed a database of iriscodes on the EVM. We built this database in an off-line fashion using all of our same project code, by just uncommenting one line in the PC-side code that simply appended the EVM-computed iris code to a database file on disk on the PC (or created a file if one didn't already exist.) We ran this code with the uncommented line 15 times using 15 different eye images, thereby building up our database of valid users. This database was transferred to the EVM on program startup, just after the Gabor filters are transferred, and before any interaction with the user. The time to transfer is not noticeable to the user.

The EVM compares the sample iris code against every iris code in the database, by computing the Hamming distance (see Equation 8).

$$HD = codeA \quad codeB$$

Equation 8 - Equation for computing the Hamming distance between two codes

The EVM keeps a list of all computed Hamming distances as well as the index of the best match, i.e., lowest Hamming distance. The EVM sends the entire list of Hamming distances, as well as the index of the best match, back to the PC. (The EVM also sends the computed sample iris code back to the PC, but this is only used for database building and data analysis purposes.)

Once the PC has the list of Hamming distances and the index to the best match from the EVM. It simply applies an accept criterion, originally set to 0.25, to see if the sample iris code and the best match in the database differ by fewer than that fraction of bits. If so, the PC reports that the match is good and the user is identified. If not, the PC reports that the match is not reliable and the user is rejected. This method is equivalent to saying that a sample iris code passes or fails a test of statistical independence when compared against all other iris codes. The validity of this approach is discussed below.

In emailed dialogue with Daugman, he let us know that he uses a criterion of 0.32 and then automatically scales this back as the size of his database grows and the probability of error accumulates. In retrospect, 0.32 would have been a better value for us as it would have reduced our number of false negative by 2, and would not have affected our perfect record of zero false positives since at no time did unrelated irises differ in any fewer than 40% of their bits.

Discussion on Hamming Distance and the Test of Statistical Independence

Although it is quite clear on simple observation that there is manifest correlation between unrelated irises in general anatomic form and physiology, it is perhaps less clear that the actual texture of the iris and the minutiae that form it are stochastic. To wit, the iris naturally comes with a certain number of degrees of freedom associated with it, at specific, small scales of analysis. The method of testing the statistical independence of two iris codes is based on this innate randomness at these scales of analysis.

In extracting this randomly generated information contained within the structures of the iris, we have to be cognizant of two factors that will reduce the ideal number of degrees of freedom, and thus the information capacity of our iris code. First, the method of extraction, in our case phase information extraction using Gabor wavelets, reduces information capacity by introducing coherence. In the case of our specific filter, Daugman computes that iris code capacity is reduced by a factor of 4.05 due to this coherence. (This number does not directly translate to our project since it depends on parametric values and sampling densities used by Daugman that are unknown to us, but it is at least a ball park figure.) Second, inherent correlations within the iris will reduce the information capacity of our iris code as not all bits will be independent. Extensive study by Daugman of information capacity and Hamming distance distribution across a sampling of 9.1 million iris codes strongly suggests that his 2048-bit codes carry the equivalent of 249 degrees of freedom. Additionally, Daugman has found that each bit in any given iris code and across the population of iris codes is equally like *a priori* to be 0 or 1. With these two well-supported empirical observations in hand, we can see that the randomness expressed in Daugman's iris code translates into 249 flips of a fair coin. Daugman's testing confirms this binomial distribution.

In our case, the iris code is approximately equivalent to 31 flips of a fair coin. And so we can expect the likelihood of two of our iris codes from different eyes agreeing completely by chance to be roughly 1 in 2^{31} or 1 in 2,147,483,648. Further, we can calculate the odds that a sample iris code will be falsely accepted, using our 0.25 criterion, against another unrelated iris code, simply by chance to be roughly 1 in 272, using Equation 9 where N = degrees of freedom (31), m = number of matching degrees (8), $p = q = 0.5$, and x is our fractional Hamming distance (m / N).

$$f(x) = \frac{N!}{m!(N - m)!} p^m q^{(N - m)}$$

Equation 9 - Fractional function form of the binomial distribution

Although by no means do we have a formal proof of this method of comparison, we are supported by a strong and well-understood statistical framework and a wealth of confirming empirical data. In addition, it is interesting to note that all commercially available iris recognition products (all licensed from Daugman) currently employ this test of statistical independence between iris codes as their method of acceptance and rejection.

Testing and Results

Since our database of eye images only contained one image per eye, we were clearly lacking the variability necessary to test our implementation. So, we took the step of introducing our own variability. In essence, we screwed up the images a little to see how our system would react. We decided on two kinds of modifications for our images. First, we introduced a Gaussian blur (setting of 50 in Irfanview) to each of the 48 images in our database, saving each with "_b50" appended to its filename. Second, we introduced obstructions to each of our 48 database images by drawing on the images using a paint program, in effect, mimicking a dirty camera lens or eyelashes covering a portion of the iris. We saved each of these obstructed images with "_o" appended to its filenames. So, our final test set consisted of 144 total eye images, 48 untouched images, 48 blurred images, and 48 obstructed images. We tested each of these 144 images against our database of 15 authorized users (see example below).

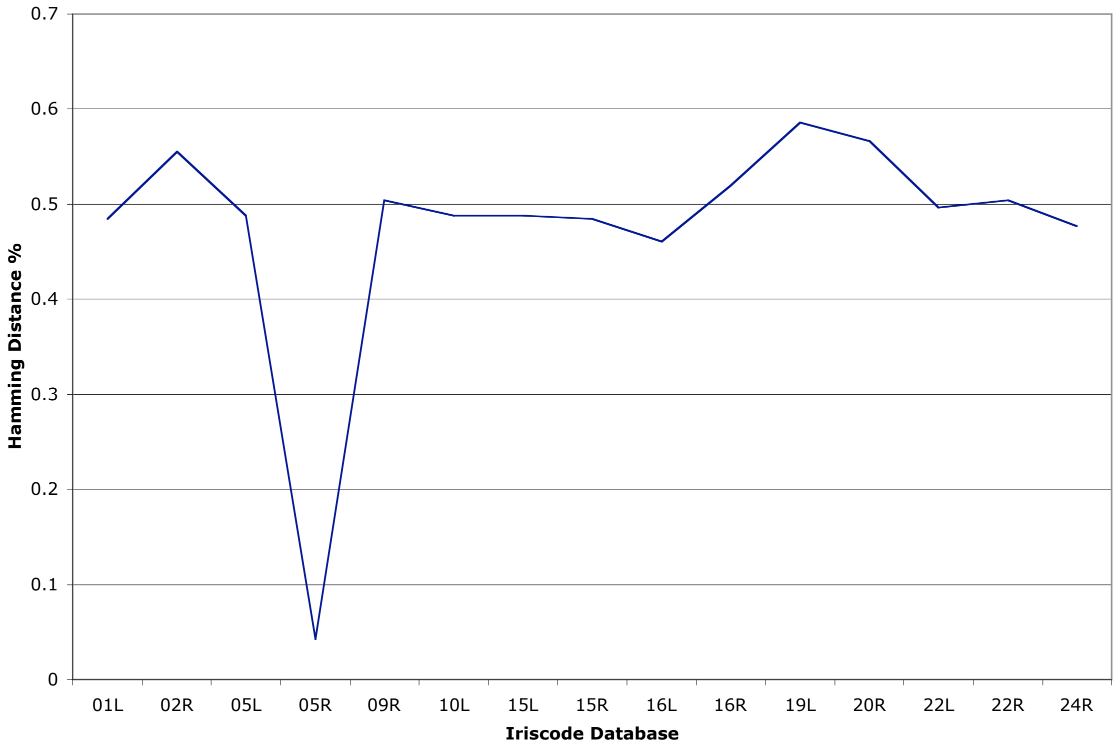


Figure 3 - Graph of Hamming Distance for Image 05R with Blurring

In no test did we receive a false positive; we never authorized anyone that we shouldn't have. In fact, in our tests, the fractional Hamming distance between two distinct irises never fell below 40%. We did receive 4 false negatives (see below) in which we erroneously rejected authorized users. Each of these 4 false negatives occurred when the sample image came from the "obstructed" or "blurred" class of images. Our Hamming distance between identical, unaltered images was always 0 as expected. Interestingly, in all cases of erroneous rejection, the closest matching user, although outside our threshold, was always the proper user. In our worst miss, for the obstructed image "20r_o", the reported fractional Hamming distance was higher than the lowest Hamming distance reported between unrelated irises in other tests. In other words, there was no threshold that we could set that would properly accept "20r_o" and still record 0 false positives in all other tests.

Image Name	Alteration	Hamming Distance
10 L	Blur	.378
15 R	Blur	.266
20 R	Obstruct	.445
22 L	Blur	.332

In almost all cases, the iriscode died gracefully, reporting small Hamming distances for small modification to images, and larger Hamming distances for larger disturbances.

In examining the log files for the runs in which erroneous results were reported, it became clear that in most cases, there was a failure by the iris location algorithms to properly locate the boundaries of the iris. It is not entirely clear why the algorithm failed in certain cases, although we had long suspected that the iris-sclera boundary in some images was too close to the image edge for effective convolution with the blurring function.

In one case, we could not attribute the failure to the location software. In this case, the reported Hamming distance seemed to be slightly out of proportion to the distances reported for similar blurring in other images. It is not clear if this is just a natural variation in the system or an implementation error.

Speed and Memory Allocation

Transfers from the PC to the EVM dominate the trivial transfers in the other direction. At program start up, we send two chunks of data to the EVM, the iris code database, and the Gabor filters. Additionally, for each accept/reject test that is run, a single iris image is sent to the EVM. Each of these three data components is small relative to the capabilities of the various buses between the PC and the EVM. So, we decided to use HPI transfers, lab 3 code, rather than PCI since our experience has shown that the former, although slightly slower, are far more reliable for our purposes.

Item	Memory Location	Transfers	Size
Iris code Database	On Chip	HPI Transfers	~6K
Gabor Filters	On Chip	HPI Transfers	~3.5 K
Iris Image	On Chip	HPI Transfers	~30 K

In profiling our code, we saw that computing the iris code from an image was by far the most computationally intensive part of our code. Computing the code for a single iris took some 14 million-clock cycles. Further, we saw that because of our large heap size and the ".sysmem > SDRAM0" designation in the CMD file, all of our malloc calls were returning off-chip memory. This meant that the iris image and the Gabor filters were all being stored in slow memory. As an improvement, we decided to fit as much of our data as possible in on-chip memory, paging if necessary. Fortunately, by limiting our heap size, and changing the .sysmem call to point to ONCHIP_DATA, we were able to fit all of our data in on-chip memory, and achieve significant speed improvements (see below).

Optimizer level	**Number of Cycles**
0 (None)	7.72 million
1	3.85 million
2	3.36 million
3	3.29 million

Future work

In a perfect world, our first improvement would be to develop or find professional-grade iris location algorithms. It was a good learning experience developing our own, but we need to learn a little more. In particular, we could never get the iris-pupil boundary detection working in a reliable way, and this would need to be rectified first.

A further, rather simple improvement would be to increase the numbers of segments we break the iris into for analysis and increase the number of scales of analysis. This would bring us closer to the amount of information that Daugman extracts from the iris, and increase our reliability to something on the order of his.

Additionally, we would have liked to incorporate the ability to tolerate slight rotations in the eye image. (In testing, we determined that rotating an image 5^o renders it statistically independent from its unrotated version.) Daugman describes a method of doing this in which a sample iris code is compared many times to each code in the database, each time rotating its bits in a cyclic fashion.

Sensitivity analysis could also be included. Some areas that could be studied are the resolution necessary to maintain reliability and the amount of obstructions that could be introduced. Reducing the resolution of the image would allow for quicker transfer rates. However, there is a tradeoff between the increase in speed and the decrease in reliability. Sensitivity analysis could show where the tradeoff becomes unacceptable. Additionally, additional obstructions could be added to determine at what level of obstruction the system looses reliability. For both of these analyses, the desired level of reliability would need to be defined. This could change depending on the user and the purpose of the screening.

The Demo

Our demo begins by welcoming the user, displaying the authorized scans, and prompting the user to enter the name of an image. Image names are based on the user number and designation of which iris the scan is. For example, the left iris can from person 24 is named 24l. The user can select any iris can in our image database. For the purpose of our demo, we are treating each image as a different user. Once an image name is inputted, the user is then prompted to enter any desired alterations: obstruct, blur or none. If the selected scan is that of an authorized user, an acceptance is posted along with the identity of the user and the Hamming Distance of the best match. If the selected scan is not that of an authorized user, the user will be rejected but the closest match and associated Hamming Distance will be listed. The user is then given the option to submit another image or quit. When the user quits, a log file is printed of the entire session. This includes the iriscodes for the authorized users, the calculated iriscode for each test scan and the Hamming Distance comparisons to each authorized user. A sample log file is included in Appendix A.

References

Daugman, John. “How Iris Recognition Works.” http://www.cl.cam.ac.uk/users/jgd1000/

This reference provided us with our starting point and the general equations used in the project. However, while providing a good overview, this paper did not include a lot of in depth information.

Group 3, 18-551, 1999. “A Study in Iris Recognition, Final Report.” http://www.ece.cmu.edu/~ee551/Old_projects/projects/s99_3/final.html

This reference provided much of the in depth information we needed including many of the formulas that we used. Our goal was to improve on the work done by this group four years ago.

University of California at Santa Barbara. http://vision.ece.ucsb.edu/software.html

This is the only C code for Gabor that we found on the web. However, it did come with a warning to use at your own risk. The documentation was almost non-existent. We had difficulties determining what form the input and output would be required to be in. Therefore, we decided not to use this code.

Appendix A

Sample Log File

```
Authorized Users Database:

Name      IrisCode
01l
110110000110010011000110110110001011010010101111001000010010111111111001
111101111101101101110100111011111101101111000111111001000100100000001011
101111011101101100111001101100110101100101100101010011010101110111001111
00100100100011101011101011000000010010010
02r
101000111011001100011110000000000101001100000100010100100000000100010110
101000100101000011001011100100011101011110011111001111000111101010010110
010010010001010101011010100001110111101000100001111100000111011111000011
10001101101100110111011100110101011111000
05l
100111010110000111001100000101010100110100001000010011111100101000001111
100010000000110110010000010011011111100101001111011000010010110001110100
101101110001110010100101110101100000010100001000010001101011010000100110
10111000000100101001110101010111011000000
05r
010000110011111001001001111100011100100110111001111010101101100101001010
110110111110100011110001010010001111000101100000010011000000001001110011
110001011000110101100001001010010000111011100101000111000111110100001101
01111100000010011111000011001100010100 11
09r
100011001110111101111011111010111111001011001011101010101101000011110110
000001000101011000000111110001001100110011000111110011100010111011010011
101001000010100001010100010110010101011010010000111110010000000100100000
00000101100101010101010111111010000010000
10l
000000001001100110110101100100011110111010010001100000011001100101001001
010100110001101000101010110010100111110011101111101100110101101010010111
111101111110001111101111100100010000010100011000111010011011111000000100
10111110100110011100110111000111001111111
15l
011100110111111101110111111100101000010110101111001111010000011010000000
000110110111000110011110111010101011100110011001111110000101101111001111
011111001101101000011001100011100000111010000000110010000100010110111011 0
10100011000000111101000000001011010111000
15r
001010111011101000101001111001011101111011000010100111000001101010101100
011100100111110011111010001101101111111001100101010110000011001100100010
011100110000001111011100010000110111110000000100101000111000111001010100 00
```

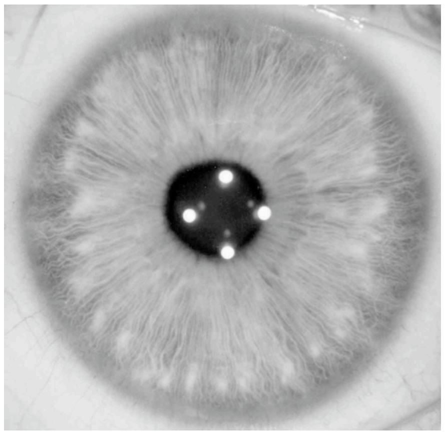
```
1100001001111101111001001001111111001101
16l
110010000111010101000000010101110100100001110100100111001100010111011011
010001000101001111000000100001010111110110111111010011100110111100110001
110010010110101111001101100001001001111111000110100011110010011111100111
0000010100010011101101011011111110100001
16r
110110101100101011011110111111101101101011001011110010100111011010010100
111100001111001001110110001000010001101000011011000101100101101010111000
110111100110111010010011000011011100011000001001101010111110100110010100
1100001110101111111000000100001011101011
19l
110111111011000010100100011110000001110011111010001111010111111010001101
101011000100101110101100001001111010010010111001001001101011000010101001
011100001101111100111010010110101111010101100000001010110000100111101001
1010100110100100001000001111001000100110
20r
001010011001100101111110101111011111101111010111011010110001011011110001
001011101100110101001100000111110100111111001011010111111110001110101111
010100011001001111010010110011101110100010010010111100011010010101111111
1010110011111111111011101101000010000011
22l
001000100001000000001001010011101001000101001010011001010001011011111110
011001000100111001000100010010100000100011101000111010000000100011111001
101010110100100100011010010111101000010011101011010100000110101100100110
0101110010101000100000001101000001100100
22r
001010001011101100011011110010010010010111111111000100100101010010001100
000000100111111000100010011010000011011000010001100111110011100010111101
101111110010010000111110111110001101010111010100100101110110110100000101
0001000000100100011001111100000011101010
24r
001010000110010000111011111100011101101100111000111000110100001011011011
011001101110110111100011111111100101011110001101010000110110101001100110
101011111110000000111100001110011011000111010111101111111001111010101000
1010000101110101111101010100100110100110

*******
Supplied Name: 05r
Supplied Image: c:\group6\project\images\05r_b50.bmp
Detected Sclera Boundary: center (187, 203)  radius 182
Detected Pupillary Boundary: center (187, 203)  radius 50
IrisCode:
010000110011111001001001111100011100100110111001111010101101100101000010
010110111110100011110001010010001111000101100000010011000000001001110011
110000001010110101100001101010010000101011000101000011000101110100001101
```

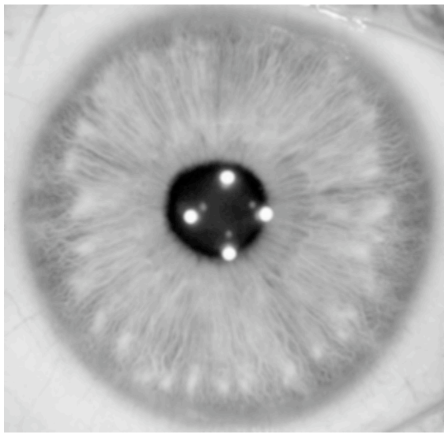
```
0111100000001001111100001100110001010011
Best Match: 05r

Sample image Vs. Database

Name   HD     HD(%)
01l (124) (48.437500)
02r (142) (55.468750)
05l (125) (48.828125)
05r ( 11) (4.296875)
09r (129) (50.390625)
10l (125) (48.828125)
15l (125) (48.828125)
15r (124) (48.437500)
16l (118) (46.093750)
16r (133) (51.953125)
19l (150) (58.593750)
20r (145) (56.640625)
22l (127) (49.609375)
22r (129) (50.390625)
24r (122) (47.656250)
*******
Supplied Name: 05l
Supplied Image: c:\group6\project\images\05l_o.bmp
Detected Sclera Boundary: center (193, 192)  radius 188
Detected Pupillary Boundary: center (193, 192)  radius 50
IrisCode:
1001110101100001110011000001010101011101000010000100001101001010010001111
1010101000001101100000100100110111111001010011110110000100101100011101000
1011011100011100101001011101011000000101000010000100111010110101001001100
1011100000100101001110101010111011000000
Best Match: 05l

Sample image Vs. Database

Name   HD     HD(%)
01l (123) (48.046875)
02r (129) (50.390625)
05l ( 12) (4.687500)
05r (124) (48.437500)
09r (132) (51.562500)
10l (126) (49.218750)
15l (140) (54.687500)
15r (135) (52.734375)
16l (123) (48.046875)
16r (140) (54.687500)
```

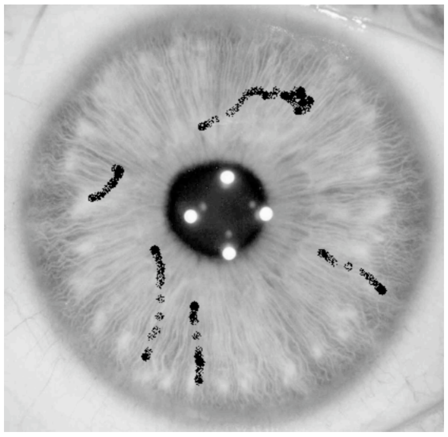
19l (129) (50.390625)
20r (134) (52.343750)
22l (132) (51.562500)
22r (136) (53.125000)
24r (121) (47.265625)

Appendix B

Sample Images


Original Image


Blurred Image


Obstructed Image